#4



Docket No.: P-0311 PATENT

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

| | | |
|---|---|---|
| In re Application of | : | |
| Hyun Duk CHO and Sung Deuk KIM | : | |
| Serial No.: 10/002,404 | : | Group Art Unit: 2613 |
| Confirm. No.: 1015 | : | Examiner: Unassigned |
| Filed: December 5, 2001 | : | |
| For: VIDEO DATA CODING/DECODING APPARATUS AND METHOD | : | |

**TRANSMITTAL OF CERTIFIED PRIORITY DOCUMENT(S)**

Assistant Commissioner of Patents
Washington, D. C. 20231

Sir:

At the time the above application was filed, priority was claimed based on the following application(s):

Korean Patent Application No. 73695/2000 filed December 6, 2000

A copy of each priority application listed above is enclosed.

Respectfully submitted,
FLESHNER & KIM, LLP

Daniel Y. J. Kim
Registration No. 36,186

P. O. Box 221200
Chantilly, Virginia 20153-1200
703 502-9440 DYK/cah
**Date: February 26, 2002**

# 대 한 민 국 특 허 청
# KOREAN INTELLECTUAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Intellectual Property Office.

출 원 번 호 : 특허출원 2000년 제 73695 호
Application Number PATENT-2000-0073695

출 원 년 월 일 : 2000년 12월 06일
Date of Application DEC 06, 2000

출 원 인 : 엘지전자주식회사
Applicant(s) LG ELECTRONICS INC.

2001 년 11 월 21 일

특 허 청
COMMISSIONER

【서지사항】

| | |
|---|---|
| 【서류명】 | 특허출원서 |
| 【권리구분】 | 특허 |
| 【수신처】 | 특허청장 |
| 【참조번호】 | 0001 |
| 【제출일자】 | 2000.12.06 |
| 【국제특허분류】 | H04L |
| 【발명의 명칭】 | 데이터 분할 기법을 이용한 영상 부호화 전송방법 |
| 【발명의 영문명칭】 | A METHOD FOR VIDEO CODING AND TRANSMITTING USING DATA PARTITIONING METHOD |
| 【출원인】 | |
| 【명칭】 | 엘지전자 주식회사 |
| 【출원인코드】 | 1-1998-000275-8 |
| 【대리인】 | |
| 【성명】 | 허용록 |
| 【대리인코드】 | 9-1998-000616-9 |
| 【포괄위임등록번호】 | 1999-043458-0 |
| 【발명자】 | |
| 【성명의 국문표기】 | 조현덕 |
| 【성명의 영문표기】 | CHO,Hyun Duk |
| 【주민등록번호】 | 681112-1055412 |
| 【우편번호】 | 463-070 |
| 【주소】 | 경기도 성남시 분당구 야탑동 339 장미마을 834동 1403호 |
| 【국적】 | KR |
| 【발명자】 | |
| 【성명의 국문표기】 | 김성득 |
| 【성명의 영문표기】 | KIM,Sung Deuk |
| 【주민등록번호】 | 720928-1795811 |
| 【우편번호】 | 431-080 |
| 【주소】 | 경기도 안양시 동안구 호계동 993-3 유환아파트 1동 105호 |
| 【국적】 | KR |
| 【심사청구】 | 청구 |

【취지】 특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다. 대리인 허용록 (인)

【수수료】

| | | | |
|---|---|---|---|
| 【기본출원료】 | 20 | 면 | 29,000 원 |
| 【가산출원료】 | 5 | 면 | 5,000 원 |
| 【우선권주장료】 | 0 | 건 | 0 원 |
| 【심사청구료】 | 7 | 항 | 333,000 원 |
| 【합계】 | 367,000 원 | | |

【첨부서류】 1. 요약서·명세서(도면)_1통

# 【요약서】

## 【요약】

본 발명은 데이터 분할(Data Partitioning) 기법을 이용한 동영상 압축 부호화를 기반으로 하여, 각 파티션(Partition) 별로 채널 부호화(Channel Coding)를 수행함으로써 보다 중요한 정보에 대해서 강한 에러 보호 능력을 부여할 수 있도록 하고, 에러가 심한 전송 환경에서 동영상을 전송할 때, 중요한 정보의 손실이 상대적으로 줄어들도록 하여 보다 좋은 화질로 동영상을 전송할 수 있도록 한 영상 부호화 전송방법에 관한 것이다.

본 발명은 데이터 분할 기법(Data Partitioning Method)을 기반으로 하여 동영상 데이터를 부호화 및 전송하는 시스템에 있어서, (a). 소스 코딩된 동영상 데이터를 소정의 블록 사이즈로 구분하는 단계, (b). 상기 소정의 블록 마다에 해당하는 채널 코딩 바이트 수를 산출하는 단계, (c). 상기 해당 블록 마다 상기 산출된 바이트 수의 채널 코딩 비트를 첨가하는 단계, (d). 상기 채널 코딩율을 각각의 파티션 별로 다르게 적용하여 비트 스트림을 전송하는 단계; 로 이루어진 것을 특징으로 하는 데이터 분할 기법을 이용한 영상 부호화 전송방법이다.

【대표도】

도 2

【색인어】

동영상 부호화, 동영상 전송

【명세서】

【발명의 명칭】

데이터 분할 기법을 이용한 영상 부호화 전송방법{A METHOD FOR VIDEO CODING AND TRANSMITTING USING DATA PARTITIONING METHOD}

【도면의 간단한 설명】

도1은 H.263++ 데이터 분할 기법을 설명하기 위한 비트 스트림 구조를 나타낸 도면

도2는 본 발명 제1실시예로서, 데이터 분할 정보 테이블을 이용한 데이터 분할 리드-솔로몬 부호화 방법을 설명하기 위한 비트 스트림 구조를 나타낸 도면

도3은 본 발명 제2실시예로서, 마커를 이용한 데이터 분할 리드-솔로몬 부호화 방법을 설명하기 위한 비트 스트림 구조를 나타낸 도면

도4는 본 발명에서 마커 에뮬레이션 검색 기법을 설명하기 위한 데이터 구조의 예를 나타낸 도면

도5는 데이터와 마커 사이의 에뮬레이션 발생을 설명하기 위한 비트 스트림 구조를 나타낸 도면

도6은 본 발명에서 정보비트와 잉여비트의 치환을 이용한 마커 에뮬레이션 제거 기법을 설명하기 위한 도면

도7은 본 발명의 제2실시예에 따른 비트 스트림에서 파티션1(헤더) 및 파티션2(움직임 벡터)의 데이터 부호화 구조의 예를 나타낸 도면

도8은 본 발명의 제2실시예에 따른 비트 스트림에서 파티션3(DCT 계수)의 데이터 부호화 구조의 예를 나타낸 도면

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<9> 본 발명은 동영상 데이터를 압축 부호화하여 전송하는 방법에 관한 것이다.

<10> 특히 본 발명은 데이터 분할(Data Partitioning) 기법을 이용한 동영상 압축 부호화를 기반으로 하여, 각 파티션(Partition) 별로 비등가 에러 보호(Unequal Error Protection)를 수행하기 위한 부호화 방법으로서, 각 파티션 별로 채널 부호화(Channel Coding)를 수행함으로써 보다 중요한 정보에 대해서 강한 에러 보호 능력을 부여할 수 있도록 하고, 에러가 심한 전송 환경에서 동영상을 전송할 때 중요한 정보의 손실이 상대적으로 줄어들도록 하여 보다 좋은 화질로 동영상을 전송할 수 있도록 한 영상 부호화 전송방법에 관한 것이다.

<11> 동영상을 압축 부호화하여 전송하는 시스템에서 데이터 분할 기법을 이용한 동영상 압축 부호화 전송기법이 제안되고 있다.

<12> 도1은 H.263++ 에서 데이터 분할 기법을 설명하기 위한 비트 스트림(bit stream)의 구조를 보여주고 있다. 이 비트 스트림은 n번째 슬라이스(Slice #n)에 대한 구조를 보여준다.

<13> 비트 스트림의 선두에는 슬라이스 시작 코드(Slice Start Code)(101)가 온다. 이 슬라이스 시작 코드(또는 GOB Start Code)에 이어서 파티션 1(Partition

1)(102)이 오는데, 이 파티션1은 헤더(Header)이다. 파티션1에 이어서 파티션 사이의 경계를 표시하기 위한 마커1(Marker 1)(103)이 오고, 그 다음에 파티션 2(104)가 오는데, 이 파티션2는 움직임 벡터(Motion Vector) 정보이다. 그리고 다시 파티션의 경계를 표시하기 위한 마커2(105)에 이어서 파티션3(106)이 온다. 파티션3(106)은 DCT 계수(DCT Coefficients)이다. 그리고 슬라이스n의 후미에 바이트 얼라인(byte align) 데이터(107)가 오는데, 이 바이트 얼라인 데이터는 제로 비트를 삽입한다(Zero-bit insertion for byte-align).

<14> 상기한 바와같이 데이터 파티션 기법을 이용한 영상 데이터의 압축 전송시에는 데이터 파티션 사이에 마커를 두어 파티션 사이의 경계를 구분하고 있다. 여기서 각각의 파티션들은 데이터 비트 스트림의 선두에 오는 파티션이 후에 오는 파티션 보다 상대적으로 더 중요한 의미있는 정보가 되도록 스트리밍된다. 이렇게 분할하여 부호화하게 되면 한개의 파티션이 채널 에러로 인해서 손상되어도 나머지 파티션들의 정보를 이용해서 가능한한 원 영상과 가깝게 영상 데이터를 복원할 수 있는 잇점이 있다.

<15> 예를 들어, 데이터 전송중에 파티션3(106)이 손상되었다면 더 중요한 정보를 가지고 있는 앞선 파티션1(102) 및 파티션2(104)의 정보를 이용해서 영상을 복원하고, 파티션2(104)가 손상되었다면 파티션1(102)의 정보를 이용해서 복원한다. 그러나 파티션1(102)이 손상된 경우에는 같은 GOB 또는 슬라이스 내의 정보로는 복원할 수 없다. 그러므로 파티션1(102)은 가능한한 손상을 입지 않는 것이 좋고, 마찬가지로 상대적인 중요도가 더 높은 파티션2(104) 역시 파티션3(106)에 비해서 손상을 입지 않는 것이 좋을 것이다.

<16> 상기한 바와같이 데이터 분할 기법을 이용한 동영상 압축 전송방법은 동영상 정보를 파티션을 묶음으로 해서 파티션 간의 에러 전파를 막을 수 있다는 장점이 있지만, 상대적으로 중요한 파티션 정보를 에러로부터 근본적으로 더 잘 보호하기는 어렵다.

<17> 따라서, 데이터 분할 기법을 이용한 동영상 압축 전송시스템에서, 더 중요한 파티션일수록 에러로부터 더 잘 보호하고, 이를 통하여 동영상 전송의 품질을 향상시킬 수 있고 수신되어 복원된 동영상의 화질을 높일 수 있는 방법이 필요하다.

【발명이 이루고자 하는 기술적 과제】

<18> 본 발명은 데이터 분할 기법을 기반으로 하는 동영상 압축 부호화 전송방법으로서, 각각의 파티션에 비등가 에러 보호를 위한 채널 코딩을 수행하여, 데이터 파티션 영역별로 에러 보호 수준을 달리함으로써, 중요한 데이터 영역에 대해서 에러가 발생할 확률을 더욱 낮추어 줄 수 있도록 한 데이터 분할 기법을 이용한 영상 부호화 전송방법을 제안한다.

<19> 본 발명은 데이터 분할 기법을 기반으로 하는 동영상 압축 부호화 전송방법으로서, 각각의 파티션별로 다른 수준의 채널 코딩을 적용함에 있어 리드-솔로몬 코딩(Reed-Solomon Coding)을 적용하여 중요한 데이터 영역에 대한 에러 발생의 확률을 감소시킴으로써, 에러가 있는 전송 환경에서 동영상 정보를 전송할 때, 보다 좋은 화질로 전송할 수 있도록 한 데이터 분할 기법을 이용한 영상 부호화 전송방법을 제안한다.

【발명의 구성 및 작용】

<20> 본 발명의 데이터 분할 기법을 이용한 영상 부호화 전송방법은, 데이터 분할 기법(Data Partitioning Method)을 기반으로 하여 동영상 데이터를 부호화 및 전송하는 시스템에 있어서, (a). 소스 코딩(source coding)된 동영상 데이터를 소정의 블록 사이즈(block size)로 구분하는 단계, (b). 상기 소정의 블록 마다에 해당하는 채널 코딩(channel coding) 바이트 수를 산출하여 각각의 파티션 별로 채널 코딩율(channel coding rate)을 다르게 적용하여 코딩하는 단계, (c). 상기 해당 블록 마다 상기 산출된 바이트 수의 채널 코딩 비트를 첨가하여 상기 파티션 별로 다른 코딩율로 채널 코딩된 동영상 데이터 비트 스트림(bit stream)을 전송하는 단계; 로 이루어진 것을 특징으로 하는 데이터 분할 기법을 이용한 영상 부호화 전송방법이다.

<21> 또한 본 발명의 데이터 분할 기법을 이용한 영상 부호화 전송방법에서, 상기 채널 코딩에 따른 마커 에뮬레이션을 회피하기 위한 마커 에뮬레이션 제거과정을 더 수행하는 것을 특징으로 한다.

<22> 또한 본 발명의 데이터 분할 기법을 이용한 영상 부호화 전송방법에서, 상기 마커 에뮬레이션 제거는, 마커 검색을 위한 윈도우를 설정하고, 이 윈도우와 데이터 스트림의 매칭 여부에 따라 마커를 회피하기 위한 데이터를 데이터 비트에 삽입하거나, 정보 데이터 보다 앞선 위치에 채널 코드를 스트리밍하는 것을 특징으로 한다.

<23> 또한 본 발명의 데이터 분할 기법을 이용한 영상 부호화 전송방법에서, 상기 채널 코딩에 관련된 정보로서 CCRT(Channel Coding Rate Table) 정보를 부호

화기와 복호화기가 공유하고, 부호화기에서 복호화기에 CCRT 인덱스를 전송함으로써 비트량의 계산 및 채널 복호화가 이루어지도록 한 것을 특징으로 한다.

<24> 또한 본 발명의 데이터 분할 기법을 이용한 영상 부호화 전송방법에서, 상기 파티션이 헤더(header)와 움직임 벡터(motion vector) 정보 및 DCT 계수, 그리고 각각의 파티션을 구분하기 위한 마커가 개입된 비트 스트림으로 이루어 질 때, DCT 파티션의 종단에 바이트 얼라인을 위한 제로 비트들의 삽입량을 산출할 수 있도록 해당 정보 플래그를 비트 스트림에 실어서 전송하는 것을 특징으로 한다.

<25> 또한 본 발명의 데이터 분할 기법을 이용한 영상 부호화 전송방법에서, 상기 채널 코딩은 리드-솔로몬 코딩(Reed-Solomon Coding)을 적용하는 것을 특징으로 한다.

<26> 또한 본 발명의 데이터 분할 기법을 이용한 영상 부호화 전송방법에서, 상기 채널 코딩 바이트 수(A)는, A = Trunc(정보 바이트 수 ×CCRT[index]) 로 결정하는 것을 특징으로 한다.

<27> 상기한 바와같이 이루어진 본 발명의 영상 부호화 전송방법을 첨부된 도면을 참조하여 실시예로서 설명하면 다음과 같다.

<28> 도2는 본 발명의 제1실시예로서, 각각의 파티션 별로 채널 코딩된 데이터의 비트 스트림 구조를 보여주고 있다.

<29> 슬라이스 시작 코드(201)에 이어서 파티션 테이블(Partition Table)(202)이 온다. 파티션 테이블(202)에는 후에 연속되는 3개의 파티션1(203), 파티션

2(204), 파티션3(205)을 식별할 수 있도록 파티션들의 경계 정보(각 파티션의 길이 정보)를 제공한다. 그리고 각각의 파티션들은 채널 코딩(Channel coding)이 되어 있다는 점에서 기존의 데이터 분할 기법과 다르다. 즉, 파티션1(203)은 채널 코딩 비트가 첨가되어 있으며 헤더 정보이다. 파티션2(204) 또한 채널 코딩 비트가 첨가되어 있고 움직임 벡터 정보이다. 파티션3(205)도 채널 코딩 비트가 첨가되어 있고 DCT 계수 정보이다. 그리고 한 슬라이스의 후미에는 바이트 얼라인 데이터(206)가 오는데, 이 바이트 얼라인 데이터는 제로 비트(zero bits)를 삽입한다.

<30> 도2에 따르면 본 발명에서 비등가 에러 보호를 위해서 각 파티션에 채널 코딩을 적용함은 앞서 설명한 바와 같은데, 여기서 채널 코딩은 리드-솔로몬 코딩(Reed-Solomon Coding)을 적용하였다.

<31> 채널 코딩은 크게 나누어 리드-솔로몬 코딩을 포함하는 블럭 코딩(Block Coding) 방법과 콘볼루션 코딩(Convolution Coding) 방법이 있는데, 대체로 헤더인 파티션1(203)의 크기가 작기 때문에 짧은 데이터량에 유리한 채널 코딩을 채택하는 것이 좋고, 실험적으로 보아서 리드-솔로몬 코딩이 유리하다는 결론을 얻었다. 따라서, 본 발명에서 각 파티션별 채널 코딩은 리드-솔로몬 코딩이며, 이것을 DPRS(Data Partition Reed-Solomon) 부호화 방법이라고 명명한다.

<32> 도2와 같이 파티션 테이블(202)을 비트 스트림의 선두에 두고, 여기에 후에 따라오는 파티션들의 길이 정보를 줌으로써 각각의 파티션을 식별할 수 있도록 해주면 마커 에뮬레이션(Marker Emulation)을 쉽게 피할 수 있다는 장점과 함께, 복호화기(Decoder/디코더)의 구현이 용이하다는 장점이 있다.

<33> 마커 에뮬레이션 및 그 회피방법에 대해서 설명한다.

<34> 상기 도1에서 설명한 바와같이 원래 데이터 분할 코딩 방법에서는 마커를 사용하여 각각의 파티션을 구별해 주고 있는데, 이 마커 정보(데이터)는 채널 코딩을 적용하기 전의 비트 스트림의 코드 조합으로는 나올 수 없는 코드로 만들어진 것이다. 그렇지만 본 발명에 따라서 채널 코딩을 적용한 후의 비트 스트림에서는 코드 중에 마커가 아닌데도 불구하고 마커 비트가 만들어질 가능성이 있다.

<35> 그러므로 본 발명에서는 마커를 두지 않고 도2와 같이 파티션 테이블(202)에 각각의 파티션(203,204,205)의 길이 정보를 저장한다. 이렇게 파티션 길이 정보를 테이블로 저장해서 비트 스트림 선두에 주면, 마커를 두지 않고도 파티션을 구별할 수 있게 된다. 그러면 마커 에뮬레이션 자체가 회피되므로 문제가 없다.

<36> 또한, 파티션 테이블(202)을 두게 되면 복호화기에서 상기 파티션 테이블의 정보를 해독함으로써 후에 연속되는 파티션들의 경계를 구분 지을 수 있기 때문에 복호화기의 구현이 용이하게 되는 것이다.

<37> 즉, 마커를 두어 각각의 파티션들의 경계를 구분짓는 경우에는 복호화기에서는 일단 마커를 찾아낸 다음 각 파티션의 데이터를 복호화해야 하므로, 이를 위해서는 적절한 버퍼를 두어 데이터를 저장했다가 마커를 찾은 후에 다시 이 마커가 주는 파티션 경계 정보를 이용해서 해당 파티션 데이터를 버퍼에서 가져와서 복호화해야 하지만, 파티션 테이블을 두면 이러한 문제를 해결할 수 있게 되는 것이다.

<38> 그렇지만 도2와 같이 파티션 테이블(202)을 두게 되면 파티션 테이블(202)은 각각의 파티션(203,204,205)에서 발생하는 정보량(사이즈)의 최대값을 표현해 주어야 하기 때문에 파티션 테이블(202)의 크기가 마커의 경우보다 커질 수 밖에 없다는 점이 따른다(그 사이즈를 표현할 수 있는 비트 수 필요).

<39> 따라서, 압축 효율의 측면에서 이 문제를 도3과 같은 본 발명 제2실시예로서 극복하는 방법을 제안한다.

<40> 도3은 본 발명의 제2실시예로서, 마커를 이용하는 경우의 DPRS 부호화 전송 방법을 보여준다.

<41> 비트 스트림의 선두에는 슬라이스 시작 코드(Slice Start Code)(301)가 온다. 이 슬라이스 시작 코드(또는 GOB Start Code)에 이어서 파티션 1(Partition 1)(302)이 오는데, 이 파티션1은 헤더(Header)이고 채널 코딩되어 있다. 파티션1에 이어서 파티션 사이의 경계를 표시하기 위한 마커1(Marker 1)(303)이 오고, 그 다음에 파티션2(304)가 오는데, 이 파티션2는 움직임 벡터(Motion Vector) 정보이고 역시 채널 코딩되어 있다. 그리고 다시 파티션의 경계를 표시하기 위한 마커2(305)에 이어서 파티션3(306)이 온다. 파티션3(306)은 DCT 계수(DCT Coefficients)이고 이 것 또한 채널 코딩되어 있다. 그리고 슬라이스n의 후미에 바이트 얼라인 데이터(307)가 오는데, 이 바이트 얼라인 데이터는 제로 비트를 삽입한다.

<42> 본 발명 제2실시예에서는 복호화기의 구현이 도2의 제1실시예 보다 다소 불리할 수도 있지만, 소요되는 비트가 제1실시예 보다 적기 때문에 비트율이 줄거

나 화질이 향상되게 된다. 그런데, 제2실시예를 구현하려면 마커를 사용함에 따라 발생되는 마커 에뮬레이션 문제를 극복하는 것이 필요하다.

<43> 도3에서도 각각의 파티션(302,304,206)은 채널 코딩 비트가 첨가되어 있다는 점에서 종래의 도1의 구조와 달라지고, 각각의 파티션들의 채널 코딩에 리드-솔로몬 코딩 기법을 적용한다는 점에서 도2의 실시예에서와 같다.

<44> 여기서는 도2의 본 발명 제2실시예를 중심으로 하여, 각 파티션의 채널 코딩 기법으로 적용된 리드-솔로몬 코딩의 적용과, 마커 에뮬레이션을 제거하는 두 가지 방법, 그리고 복호화기 측면에서 비트 수를 계산하는 방법(파티션1 및 파티션2)과 DCT계수(파티션3)에서의 총 비트량을 계산하는 방법의 순서로 본 발명을 설명한다.

<45> 먼저, 각 파티션별 채널 코딩 기법으로 적용된 리드-솔로몬 코딩은 블록 코딩(Block Coding) 방법이므로 데이터를 블록 단위로 나누어야 한다.

<46> 만약에 파티션의 길이가 미리 정해진 블록의 크기 보다 작다면 한 개의 파티션이 한개의 블록이 된다. 예를 들자면 일반적으로 파티션1(302)은 크기가 작기 때문에 한 개의 파티션이 한 개의 블록이 된다. 또, 파티션2(304)는 움직임 벡터 정보인데, 움직임 벡터의 경우 움직임이 작은 영상에 대해서는 한 개의 파티션이 한 개의 블록이 될 수 있다.

<47> 그리고, 리드-솔로몬 코딩은 바이트(byte) 단위로 이루어지고, 바이트 단위로 나누어 떨어지지 않는 나머지 비트에 대해서는 리드-솔로몬 코딩을 적용하지 않는다.

<48> 한편, 부호화기와 복호화기는 미리 정한 CCRT(Channel Coding Rate Table)를 각각 가지고 있다. 그러므로 부호화기에서 CCRT의 인덱스(index)를 복호화기로 전송하고 그 CCRT의 인덱스에 해당하는 전송율로 부호화 및 복호화를 수행하게 된다.

<49> 여기서, 부호화기 측에서의 채널 코딩 바이트 수(A)는, A = Trunc(I × CCRT[index]) 로 정해지며, I는 정보 바이트(Information Byte)의 수이고, index는 부호화기와 복호화기가 공유하고 있는 채널 코딩율(Channel Coding Rate)을 정하기 위한 CCRT 인덱스이며, 따라서 CCRT[index]는 채널 코딩율이 될 것이고, Trunc는 나머지를 버리는 절단 연산자(truncation operator)이다.

<50> 상기한 바와같이 하여 각각의 파티션별로 리드-솔로몬 코딩이 이루어 지게 된다. 이렇게 구성된 데이터의 비트 스트림에서는 앞에서 설명한 것 처럼 마커 에뮬레이션의 문제가 있으므로, 이 문제를 다음과 같은 방법으로 회피한다.

<51> 먼저, 앞서 설명한 바와 같이 부호화기는 각 파티션의 데이터를 버퍼에 저장하고, 저장된 데이터에 대해서 상기 리드-솔로몬 코딩을 수행한 다음, 리드-솔로몬 코딩된 데이터에 대해서 마커 에뮬레이션이 일어났는지의 여부를 검색한다.

<52> 도4는 마커 에뮬레이션 검색과 제거 기법의 예를 보여주는데, (a)는 주어진 데이터 비트 스트림에서 채널 코딩 비트와 정보 비트가 연결되는 부분을 도식화한 것이고, (b)는 마커 에뮬레이션을 제거하기 전의 데이터 비트, (c)는 마커 에뮬레이션을 제거한 후의 데이터 비트이다.

<53> 도4와 같이 데이터 비트 스트림이 정보 비트(401)와 채널 코딩 비트(402), 정보 비트(403)의 비트 스트림일 때, 마커 에뮬레이션 검색을 위한 윈도우(404)를 비트열을 따라 이동(sliding)해 가면서 에뮬레이션 검색을 수행하고, 윈도우 매칭시 마커 에뮬레이션 회피(제거)를 위한 '1' 비트를 삽입하는 것을 보여주고 있다.

<54> 즉, 마커 비트에서 LSB가 2비트 부족한 윈도우를 만드는데 예를 들어, 마커 비트가 '0000001' 이라고 한다면 윈도우는 LSB가 2비트 부족한 '00000'을 만든다. 그리고 이 윈도우(404)를 비트열을 따라 슬라이딩해 가면서 윈도우와 매칭되는 부분을 검색한다.

<55> 이렇게 해서 윈도우(404)와 매칭되는 부분이 나오면 그 매칭되는 비트의 뒤에 마커 에뮬레이션 회피를 위한 1비트를 삽입한다. 예를 들면 '1'을 삽입하면 된다. 이렇게 하면 채널 코딩된 비트 스트림에서, 특히 채널 코딩 비트와 정보 비트의 접합 부분에서, 마커가 아님에도 불구하고 마커 비트가 만들어지는 것을 방지할 수 있게 된다.

<56> 한편, 복호화기에서는 수신된 데이터 비트 스트림에서 마커를 찾는 것과 동시에 마커 에뮬레이션 제거를 위해 상기한 바와같이 삽입된 비트를 제거한다.

<57> 앞에서 설명한 것 처럼 마커 비트가 '0000001'이라면, 복호화기에 들어오는 비트를 검색해서 '000001'이 검색되면 이 것은 마커 에뮬레이션 회피를 위해서 변형된 데이터('1'비트가 삽입된 데이터 비트)이므로 이 데이터(000001)의 후미에 있는 '1'을 제거하면 된다.

<58> 그러나, 복호화기에 들어오는 비트를 검색해서 '000001'이 아닌 경우에는 검색중에 상기한 마커 비트(0000001)가 나타난 곳을 마커로 판정하면 되는 것이다.

<59> 위에서는 데이터 내에서 채널 코딩 비트와 정보 비트 사이에서의 마커 에뮬레이션을 다루었다.

<60> 그런데, 도5는 데이터(501)와 마커(502) 사이의 에뮬레이션 발생 가능성을 보여주고 있다. 즉, 도4의 경우는 데이터에 대해서만 고려하였기 때문에 도5와 같이 데이터와 마커 사이에 마커 에뮬레이션이 발생하면 복호화기에서는 도4의 기법으로는 마커를 찾지 못하게 된다.

<61> 도5와 같이 데이터(501)와 마커(502) 사이에 마커 에뮬레이션이 일어나게 될 때 마커까지 포함하여 도4의 기법을 적용하면 마커 중간에 비트 삽입이 이루어지게 될 것이고, 이렇게 되면 마커 자체의 변형이 일어나게 되어 복호화기에서는 마커를 찾을 수 없게 된다.

<62> 도5에서는 마커가 '101000101'이고, 윈도우가 '1010001'인 경우 데이터(501)와 마커(502) 사이에 마커 에뮬레이션이 일어난 것을 보여주고 있다.

<63> 이러한 경우에 도4와 같이 '1010001' 다음에 '1'비트를 삽입하게 되면 마커 중간에 '1' 비트가 삽입되어 마커 자체가 변형되기 때문에 복호화기에서는 마커(101000101)를 찾을 수 없게 되는 것이다.

<64> 따라서, 이러한 경우에는 마커 에뮬레이션 회피를 위한 비트 삽입을 마커 직전에 해야 되는데, 이렇게 마커 직전에 비트 삽입을 하게 되면 결국 부호화기가 복잡해지고, 또 이에 따라서 복호화 하기도 어려워진다.

<65> 그러므로 본 발명에서는 이러한 경우에는 정보 비트(information bit)와 잉여 비트(redundant bit)를 치환한다. 즉, 도6에 나타낸 바와 같이 채널 코딩 비트(601)의 위치와 정보 비트(602)의 위치를 바꿔서 채널 코딩 비트가 정보 비트의 앞에 오도록 하는 것이다. 이렇게 하면 마커(603)의 코드 조합은 정보 비트(602)의 코드 조합에서 나올 수 없는 코드를 할당해서 만들어 진다는 점을 기반으로 해서 마커 에뮬레이션을 회피할 수 있게 되는 것이다.

<66> 상기한 바와같이 부호화기에서는 데이터 분할 기법을 기반으로 하는 동영상 데이터의 압축 전송시에, 각 파티션별로 리드-솔로몬 코딩 기법으로 채널 코딩된 비트 스트림을 마커 에뮬레이션 제거 과정을 거쳐서 전송하고, 복호화기에서는 상기 전송된 데이터 비트 스트림을 수신하여 복호화하게 된다.

<67> 복호화기에서는 수신된 데이터 복호화를 위해서 먼저 파티션의 총 비트량과, 정보 비트량, 그리고 부가 비트량(additional bits)을 계산하는데, 상기 도3에서 설명한 본 발명 제2실시예에 따른 데이터 전송시의 비트량 계산에 대해서 설명하기로 한다.

<68> 여기서 총 비트량은 마커를 찾기 시작해서 찾을 때 까지의 비트량인데, DCT 파티션인 파티션3은 예외로 한다(이 부분은 후술한다).

<69> 비트량 계산에 있어서, 한 개의 파티션이 한 개의 블록으로 이루어졌을 경우에는 정보 바이트 수와 부가 바이트 수를 다음과 같은 C 프로그램으로 작성된 계산방법에 따라 알 수 있을 것이다. 다음에서 'Total'은 전체 바이트 수이고, 'I'는 정보 바이트 수이고, 'A'는 부가 바이트 수이고, 'CCRT[index]'는 채널 코딩율(channel coding rate)이다.

<70> int I,A,Total;

<71> float temp1, temp2;

<72> float CCRT[];

<73> I = (int)((float)Total/(CCRT[index]+1.));

<74> temp1 = (float)Total/(CCRT[index]+1.);

<75> temp2 = (float)I;

<76> if(temp1!=temp2)

<77> I++;

<78> A = Total - I;

<79> 즉, 전체 바이트 수(총 비트량)은 앞에서 설명한 바와같이 마커를 찾기 시작해서 찾을 때 까지의 비트량으로 구할 수 있고, 정보 바이트 수는 채널 코딩율(CCRT[index])과 전체 바이트 수로부터 계산할 수 있고, 부가 바이트 수는 상기 전체 바이트 수와 정보 바이트 수로부터 계산할 수 있게 된다.

<80> 한편, 한 개의 파티션이 여러 개의 블록으로 이루어진 경우에는 다음과 같이 비트량이 계산되는데, 먼저 미리 정해놓은 블록 사이즈를 BLS라고 했을 때,

블록 사이즈(BLS)에 해당하는 블록에 대해서는 정보 비트량은 곧 BLS이고, 부가 비트량은 채널 코딩율에 해당하는 블록 사이즈 즉, Trunc(BLS ×CCRT[index])이다

.

<81> 그리고 블록 사이즈(BLS) 보다 작은 사이즈의 블록에 대해서는 다음과 같은 C 프로그램으로 작성된 계산방법에 따라 비트량을 알 수 있을 것이다.

<82> 다음에서 'Total'은 전체 비트량인데, 이 것은 앞에서 설명한 바와같이 마커를 찾기 시작해서 찾을 때 까지의 비트량이다. 그리고, 'small_total', 'small_info', 'small_addi'은 각각 블록 사이즈(BLS) 보다 작은 총 비트량, 정보 비트량, 부가 비트량이며, 'CCRT[index]'는 채널 코딩율(channel coding rate)에 해당한다.

```
<83>  int Total, small_total, small_info, small_addi, temp_total;
<84>  float temp1, temp2;
<85>  temp_total = Total;
<86>  small_total = BLS+(int)(CCRT[index]*(float)BLS);
<87>   while(temp_total>=small_total) {
<88>          temp_total -= small_total; 혹은 temp_total =
<89>                                        temp_total-small_total;
<90>  }
<91>  small_info = (int)((float)temp_total/(percent+1.));
<92>  temp1 = (float)temp_total/(percent+1.);
```

<93> temp2 = (float)small_info

<94> if(temp1!=temp2)

<95> small_info++;

<96> small_addi = small_total-small_info;

<97> 즉, 파티션을 블록 사이즈(BLS) + Trunc(BLS*CCRT[index])로 구분하고, 남은 나머지가 small_total이고, 여기에서의 정보 비트량은 small_info, 부가 비트량은 small_addi 가 되는 것이다.

<98> 앞에서는 파티션1 및 파티션2에 대한 비트량 계산에 대해서 설명하였다. 그러나 파티션3(DCT 계수 파티션)에서의 총 비트량은 다음과 같이 계산되어야 한다.

<99> 즉, 파티션1 및 파티션2에 대해서는 마커를 찾는 것과 동시에 총 비트량이 계산되지만, 파티션3의 경우에는 바이트 얼라인을 위한 제로 비트가 삽입되기 때문에, 이 부분의 비트량 계산은 파티션1 및 파티션2와는 다른 방법을 사용해야 한다.

<100> 즉, 마커를 찾는다고 해도 얼마 만큼의 제로 비트가 바이트 얼라인을 위해서 삽입되었는지를 알 수 없기 때문이다.

<101> 따라서, 부호화기에서는 DCT파티션(파티션3)의 끝에 1비트의 플래그 '1'을 강제로 삽입하고, 복호화기에서는 마커를 찾은 후에 파티션 끝의 제로 비트들과 마자막 1비트를 제거한다. 이와같이 제로 비트들과 마지막 1비트를 제거한 후의 비트량이 실제의 파티션3의 총 비트량이 될 것이다.

<102> 도7은 본 발명의 제2실시예에 따른 비트 스트림에서 파티션1(헤더) 또는 파티션2(움직임 벡터)의 부호화 비트 스트림의 구조를 보여주고 있다.

<103> 파티션1에 대해서 설명하며, 파티션2 또한 이와 같다.

<104> 도7에서 (a)는 소스 코딩된 원 데이터를 나타내고, (b)는 본 발명에 따라 채널 코딩 비트가 첨가된 데이터를 나타내며, 한 개의 파티션이 여러 개의 블록으로 이루어진 경우를 보여준다.

<105> 여기서는 파티션에 해당하는 전체 소스 코딩된 비트량을 203비트(25바이트 ×8비트 + 3비트 = 203비트), 블록 사이즈(BLS) = 10바이트, CCRT = 0.5인 경우를 예로 들었다. 10바이트 블록 사이즈(BLS)의 블록(701,702)과 5바이트의 블록(703), 3비트의 데이터 블록(704)은 소스 코딩되어 있고, 마커(705)가 있다.

<106> 여기서 채널 코딩 바이트 수(A)는 앞에서 설명한 바와같이 A=Trunc(I × CCRT[index])로 계산되므로, 10바이트의 블록 사이즈(BLS)를 가지는 2개의 데이터 유닛(701,702)에 대한 채널 코딩 데이터(706,707)의 바이트 수 A = 10bytes ×0.5 = 5bytes가 되며, 5바이트의 데이터 블록(703)에 대한 채널 코딩 데이터(708)의 바이트 수는 2바이트(=Trunc(5 ×0.5))가 되고, 3비트 데이터(704)는 채널 코딩 없이 그대로 실린다.

<107> 이와같이 채널 코딩된 데이터(b)는 도4 또는 도6에서 설명한 바와 같이 마커 에뮬레이션 방지를 위한 마커 에뮬레이션 제거과정을 거친 다음 전송될 것이다.

<108> 도8은 본 발명의 제2실시예에 따른 비트 스트림에서 파티션3(DCT 계수)의 부호화 비트 스트림의 구조를 보여주고 있다.

<109> 도8에서 (a)는 소스 코딩된 원 데이터를 나타내며, (b)는 본 발명에 따라 채널 코딩 비트가 첨가된 데이터를 나타낸다.

<110> 여기서는 파티션에 해당하는 전체 소스 코딩된 비트량을 203비트(25바이트 ×8비트 + 3비트 = 203비트), 블록 사이즈(BLS) = 10바이트, CCRT = 0.5인 경우를 예로 들었다. 10바이트 블록 사이즈(BLS)의 데이터 블록(801,802)과 5바이트의 데이터 블록(803), 3비트의 데이터(804)는 소스 코딩되어 있다.

<111> 여기서 채널 코딩 바이트 수(A)는 앞에서 설명한 바와같이 A=Trunc(I × CCRT[index])로 계산되므로, 10바이트의 블록 사이즈(BLS)를 가지는 2개의 데이터 블록(801,802)에 대한 채널 코딩 데이터(805,806)의 바이트 수 A = 10bytes ×0.5 = 5bytes가 되며, 5바이트의 데이터 블록(803)에 대한 채널 코딩 데이터(807)의 바이트 수는 2바이트(=Trunc(5 ×0.5)가 되고, 3비트 데이터(804)는 채널 코딩 없이 그대로 실린다.

<112> 그리고, 앞서 설명한 바와같이 바이트 얼라인을 위해서 삽입된 제로 비트들이 얼마만큼 삽입되었는지를 알기 위해서 강제적으로 삽입한 '1'비트 플래그(information bits terminating flag)(808), 바이트 얼라인을 위한 제로 비트(809)가 실린다.

<113> 여기서 제로 비트(809)의 수(y)는, y = 8 - (total_bit% 8) 이고, total_bit는 슬라이스 시작 코드(slice start code), 마커 에뮬레이션 방지를 위

해 삽입된 비트(bit), 플래그(808), 마커 비트를 포함하는 파티션1 및 파티션2, 파티션3의 총 비트량 이다.

<114> 이와같이 채널 코딩된 데이터(b)는 도4 또는 도6에서 설명한 바와 같이 마커 에뮬레이션 방지를 위한 마커 에뮬레이션 제거과정을 거친 다음 전송될 것이다.

【발명의 효과】

<115> 본 발명은 데이터 분할 기법 기반의 동영상 압축 부호화 전송에 있어서, 각 파티션 영역별로 에러 보호 수준을 달리함으로써, 중요한 데이터 영역에 대해서 에러가 발생할 확률을 더욱 감소시켰다. 따라서, 에러가 있는 동영상 전송 환경에서 동영상 정보를 전송할 때 보다 우수한 화질의 동영상 전송이 가능하다.

【특허청구범위】

【청구항 1】

데이터 분할 기법(Data Partitioning Method)을 기반으로 하여 동영상 데이터를 부호화 및 전송하는 시스템에 있어서,

(a). 소스 코딩(source coding)된 동영상 데이터를 소정의 블록 사이즈(block size)로 구분하는 단계, (b). 상기 소정의 블록 마다에 해당하는 채널 코딩(channel coding) 바이트 수를 산출하여 각각의 파티션(partition) 별로 채널 코딩율(channel coding rate)을 다르게 적용하여 코딩하는 단계, (c). 상기 해당 블록 마다 상기 산출된 바이트 수의 채널 코딩 비트(bit)를 첨가하여 상기 파티션 별로 다른 코딩율로 채널 코딩된 동영상 데이터 비트 스트림(bit stream)을 전송하는 단계; 로 이루어진 것을 특징으로 하는 데이터 분할 기법을 이용한 영상 부호화 전송방법.

【청구항 2】

제 1 항에 있어서, 상기 채널 코딩에 따른 마커 에뮬레이션을 회피하기 위한 마커 에뮬레이션 제거과정을 더 수행하는 것을 특징으로 하는 데이터 분할 기법을 이용한 영상 부호화 전송방법.

【청구항 3】

제 2 항에 있어서, 상기 마커 에뮬레이션은, 마커 검색을 위한 윈도우를 설정하고, 이 윈도우와 데이터 스트림의 매칭 여부에 따라 마커를 회피하기 위한 데이터를 데이터 비트에 삽입하거나, 정보 데이터 보다 앞선 위치에 채널 코드를

스트리밍하는 것을 특징으로 하는 데이터 분할 기법을 이용한 영상 부호화 전송 방법.

【청구항 4】

제 1 항에 있어서, 상기 채널 코딩에 관련된 정보로서 CCRT 정보를 부호화기와 복호화기가 공유하고, 부호화기에서 복호화기에 CCRT 인덱스를 전송함으로써 비트량의 계산 및 채널 복호화가 이루어지도록 한 것을 특징으로 하는 데이터 분할 기법을 이용한 영상 부호화 전송방법.

【청구항 5】

제 1 항에 있어서, 상기 파티션이 헤더와 움직임 벡터 정보 및 DCT 계수, 그리고 각각의 파티션을 구분하기 위한 마커가 개입된 비트 스트림으로 이루어질 때, DCT 파티션의 종단에 바이트 얼라인을 위한 제로 비트들의 삽입량을 산출할 수 있도록 해당 정보 플래그를 실어서 전송하는 것을 특징으로 하는 데이터 분할 기법을 이용한 영상 부호화 전송방법.

【청구항 6】

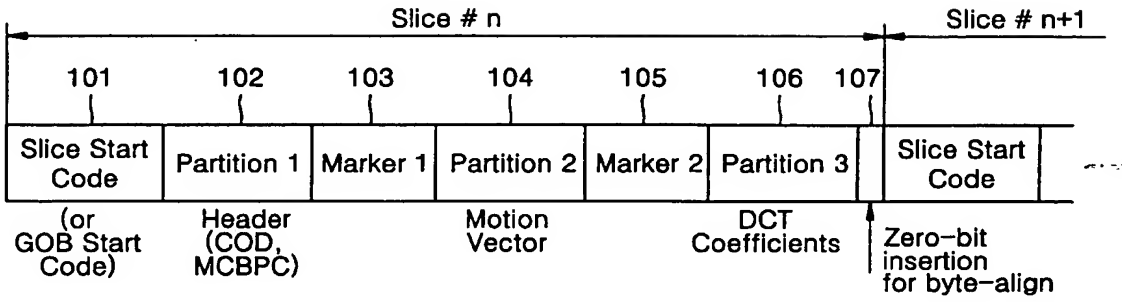
제 1 항에 있어서, 상기 채널 코딩은 리드-솔로몬 코딩을 적용하는 것을 특징으로 하는 데이터 분할 기법을 이용한 영상 부호화 전송방법.

【청구항 7】

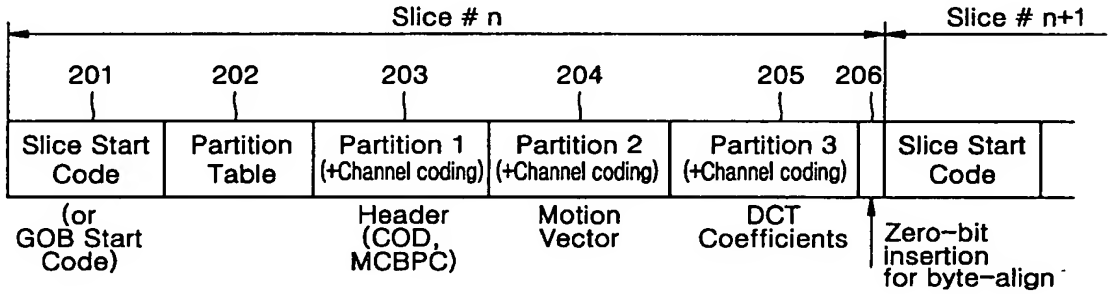
제 1 항에 있어서, 상기 채널 코딩 바이트 수(A)는, A = Trunc(정보 바이트 수 ×CCRT[index]) 로 결정하는 것을 특징으로 하는 데이터 분할 기법을 이용한 영상 부호화 전송방법.

【도면】

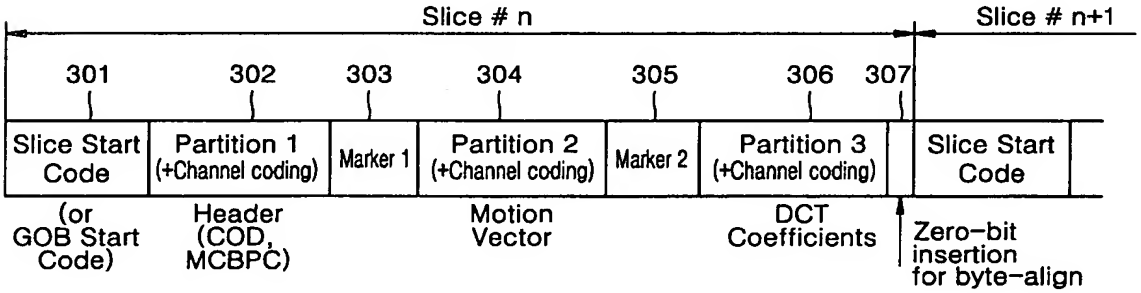
【도 1】


Slice # n
Slice # n+1
101
102
103
104
105
106
107
Slice Start Code
Partition 1
Marker 1
Partition 2
Marker 2
Partition 3
Slice Start Code
(or GOB Start Code)
Header (COD, MCBPC)
Motion Vector
DCT Coefficients
Zero-bit insertion for byte-align

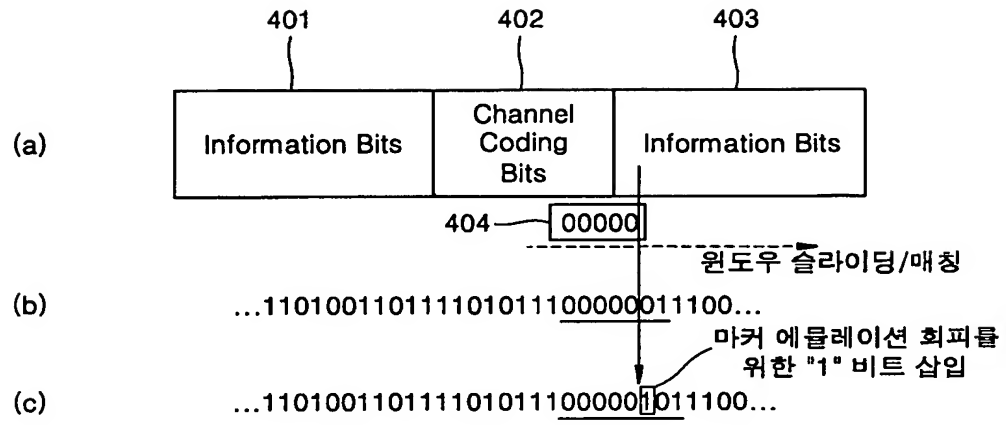
【도 2】


Slice # n
Slice # n+1
201
202
203
204
205
206
Slice Start Code
Partition Table
Partition 1 (+Channel coding)
Partition 2 (+Channel coding)
Partition 3 (+Channel coding)
Slice Start Code
(or GOB Start Code)
Header (COD, MCBPC)
Motion Vector
DCT Coefficients
Zero-bit insertion for byte-align

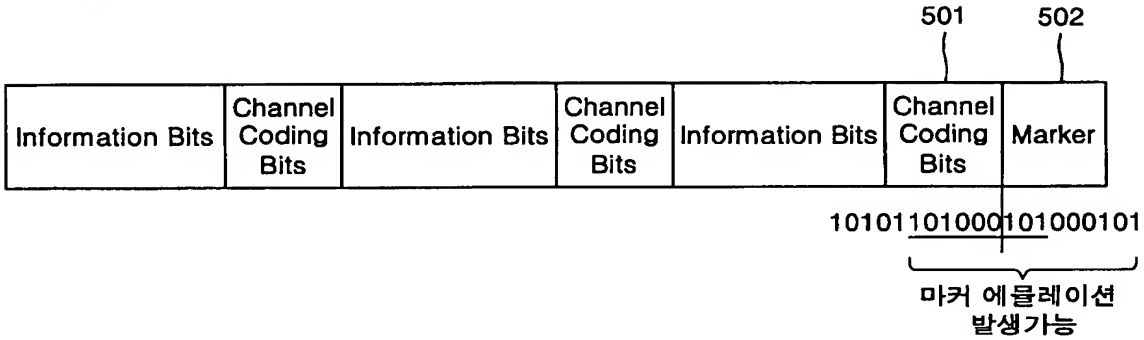
【도 3】


Slice # n
Slice # n+1
301
302
303
304
305
306
307
Slice Start Code
Partition 1 (+Channel coding)
Marker 1
Partition 2 (+Channel coding)
Marker 2
Partition 3 (+Channel coding)
Slice Start Code
(or GOB Start Code)
Header (COD, MCBPC)
Motion Vector
DCT Coefficients
Zero-bit insertion for byte-align

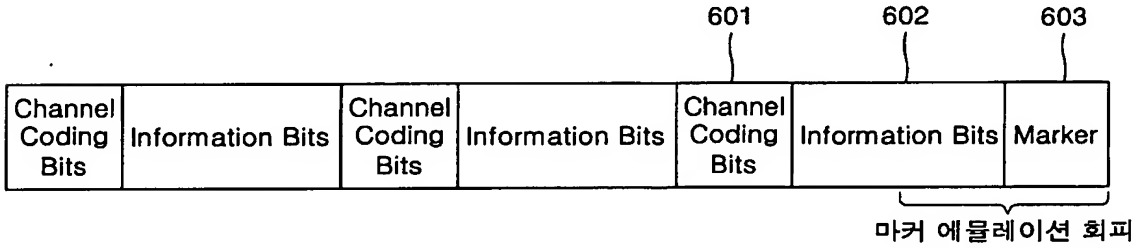
【도 4】


401
402
403
(a)
Information Bits
Channel Coding Bits
Information Bits
404
00000
윈도우 슬라이딩/매칭
(b)
...1101001101111010111000000011100...
마커 에뮬레이션 회피를 위한 "1" 비트 삽입
(c)
...1101001101111010111000001011100...

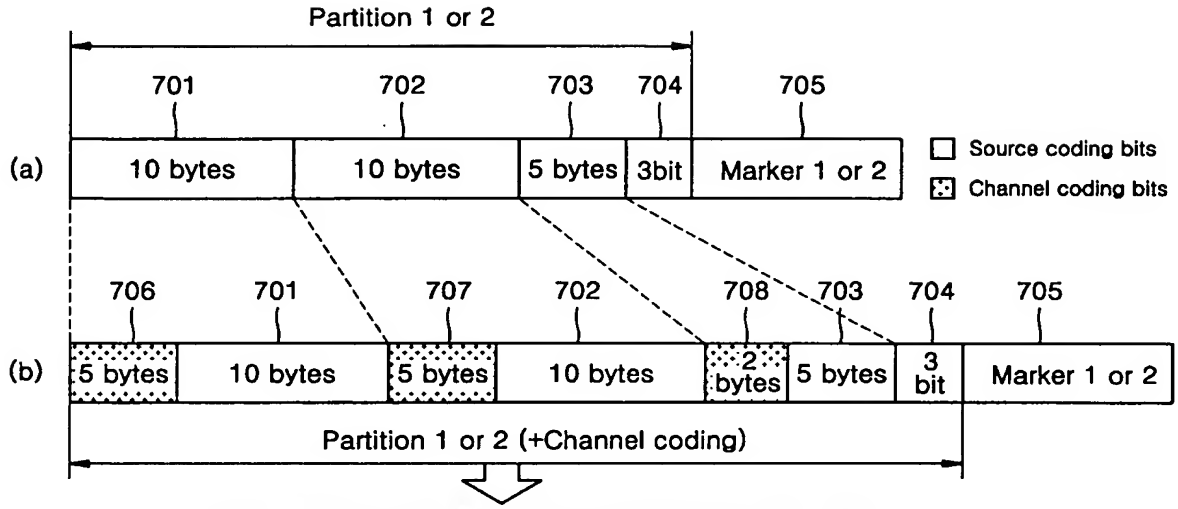
【도 5】


501
502
Information Bits
Channel Coding Bits
Information Bits
Channel Coding Bits
Information Bits
Channel Coding Bits
Marker
10101101000101000101
마커 에뮬레이션 발생가능

【도 6】

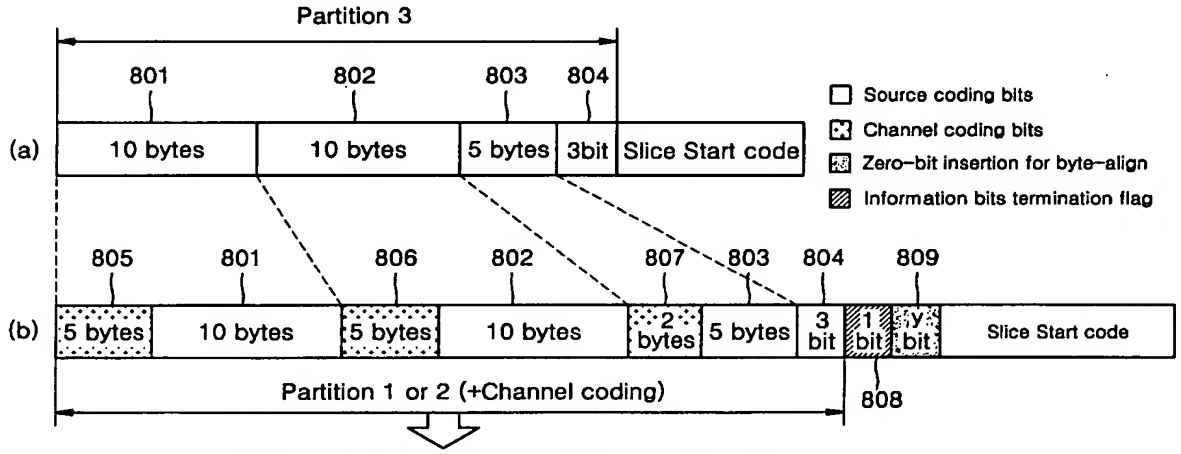

601
602
603
Channel Coding Bits
Information Bits
Channel Coding Bits
Information Bits
Channel Coding Bits
Information Bits
Marker
마커 에뮬레이션 회피

【도 7】




마커 에뮬레이션 방지를 위한 마커 에뮬레이션 제거 수행

【도 8】




마커 에뮬레이션 방지를 위한 마커 에뮬레이션 제거 수행